New horizons for Stoomboot: from Torque to HTCondor

Dennis van Dok

2024 European HTCondor Workshop, Nikhef, Amsterdam, Thu 2024-09-26

A Long-expected Party

How do you move away from a system you've been operating for over twenty years? How do you transfer two decades worth of technical proficiency to an entirely new system, while keeping the lights on? How do you get your users to make the transition with you, who are set in their ways and stand to gain little?

The Elder Days

The Physics Data Processing group (PDP) at Nikhef operates two computational facilities at some scale. Since the early 2000s we participate in the world-wide Grid, where we are committed to run the NL-T1 together with our partners at SURF.

The Stoomboot facility is a local cluster for our own researchers in physics. While smaller in scale, it can be more complicated because of a diversity of users and their workloads.

In the coming pages we will learn something about the design of the original system and how this has recently been re-implemented with HTCondor.

The Fellowship

The work described here was done by my esteemed colleagues: Mary Hester, Jeff Templon, Emily Kooistra, and Andrew Pickford. I am grateful for their hard work, the spirited discussions we had and their insightful contributions.

Many thanks alsa to the team of Miron Livny for their steadfast commitment and support.

Contents

- A Long-expected Party
- The Shadow of the Past
- Many Partings
- The Plan
- Re-imagining the queues
- Everything in a Container
- Accounting Groups
- The Express Negotiator
- Layout of nodes
- The Road Goes Ever On

The Shadow of the Past

At the time there weren't many options for running batch systems and we opted for Torque, an open source PBS derivative.

Both Stoomboot and Grid were (eventually) aligned on the same version.

The Scheduler

Stoomboot and Grid are different in scale, usage pattern and data access. The piece of software that actually matters is not Torque, but it's companion scheduler called **Maui**.

Maui

Users will almost never have to interact with Maui directly, but it controls which jobs can run on which worker nodes at what time. It decides on the prioritisation of all jobs based on a policy document that describes how we distribute the available resources to our various user (groups).

Fair Share

It is being clever about distributing the job slots in a variable supply of jobs. With the intent to let no cycles go to waste, a group *can* use more than their allocated share, at least for a while, if no other groups have jobs queued. This use-above-share is taken into account when other groups do show up, so *eventually* things should balance out.

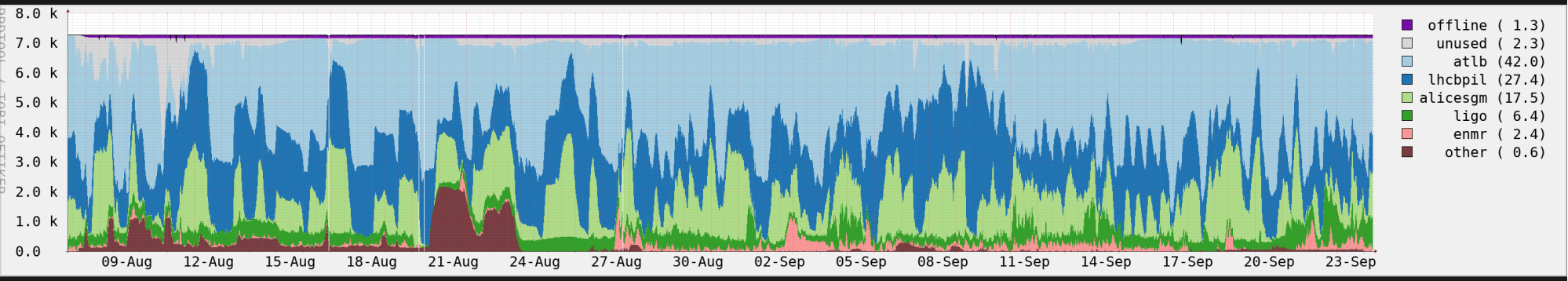
Job Roll-over

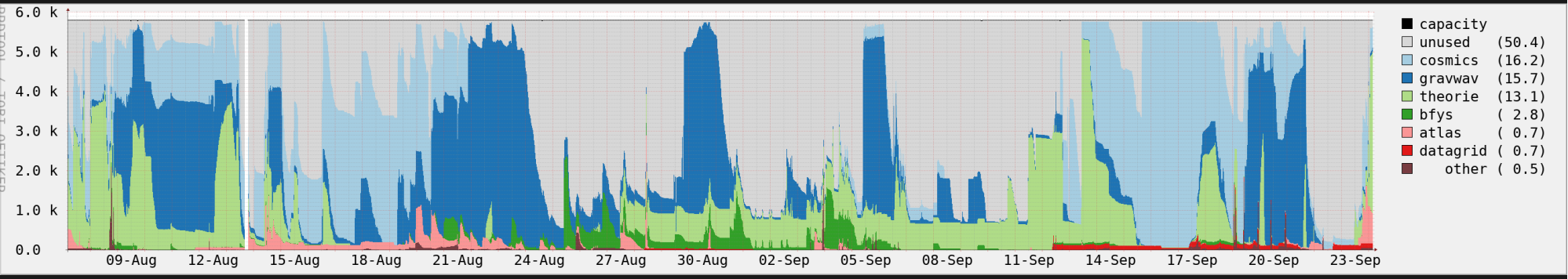
This mechanism relies on a steady rate of job slots becoming available for scheduling decisions. (To be clear: it only matters if the cluster is full, as scheduling in a cluster that is not fully occupied is trivial.)

The churn of job slots is guaranteed by the fact that a job can only last so long.

Similar, but Different

Stoomboot	Grid
Shared file system	No shared file system
Accounting by user and group	Accounting by VO
shorter jobs	longer jobs
bursty	always full


8.0 k
7.0 k
6.0 k
5.0 k
4.0 k
3.0 k
2.0 k
1.0 k
0.0
09-Aug
12-Aug
15-Aug
18-Aug
21-Aug
24-Aug
27-Aug
30-Aug
02-Sep
05-Sep
08-Sep
11-Sep
14-Sep
17-Sep
20-Sep
23-Sep
offline (1.3)
unused (2.3)
atlb (42.0)
lhcbpil (27.4)
alicesgm (17.5)
ligo (6.4)
enmr (2.4)
other (0.6)


6.0 k
5.0 k
4.0 k
3.0 k
2.0 k
1.0 k
0.0
09-Aug
12-Aug
15-Aug
18-Aug
21-Aug
24-Aug
27-Aug
30-Aug
02-Sep
05-Sep
08-Sep
11-Sep
14-Sep
17-Sep
20-Sep
23-Sep
capacity
unused (50.4)
cosmics (16.2)
gravwav (15.7)
theorie (13.1)
bfys (2.8)
atlas (0.7)
datagrid (0.7)
other (0.5)

It's the job that's never started as takes longest to finish

The Stoomboot cluster has *short*, *medium*, and *long* jobs. There is also an *express* category.

name	**maximum duration**	**availability**
express	10 minutes	immediate
short	4 hours	same day
medium	24 hours	some days
long	96 hours	maybe next week

Duration Matters

The choice for these queues is a careful balance between various interests. We could fill the entire cluster with long jobs, but that may lock up the system for the week. That is not nice for users who want to run a couple of jobs *today*.

Earmarked Hardware

We have some hardware that's been paid for by specific groups. It is part of the large shared pool and others *may* use it, but the funding group has priority. We do not preempt or kill jobs, so we only allow short jobs on these nodes from other groups; this makes the system available within 4 hours.

The Express Queue

Purely for testing purposes, the express queue is for very short duration jobs with a low maximum of total jobs.

Many Partings

- How we came to finally make the move
- What choices were made along the way

Riddles in the Dark

We learned over the past twenty years how to do things with Torque and Maui. This introduced a certain inertia where moving to anything else was going to be difficult (i.e. a lot of work) and there was hardly a reason to ever do it. The tooling around this system, from custom command-line tools to monitoring and accounting has all been developed over the many years.

Not All who Wander are Lost

With time, we found ourselves further and further from the common ground of labs that run batch systems. Torque and Maui were no longer updated (by us) and out of support. Yet everything kept working.

There was, with intervals, discussion over what software would replace it (like SLURM).

It was gradually becoming clear that HTCondor was going to be the right choice.

The Eagles are Coming

The (re-)introduction of HTCondor on Nikhef infrastructure already started in 2020; a researcher from the gravitational waves group had some grant money to spend on computers, and the plan was that we would buy and integrate this into our existing systems. It became apparent that setting up a dedicated HTCondor cluster would be a good plan.

Ganymede

This cluster ('Ganymede') was much simpler in setup than what we have now, it served only a single group and we implemented practically no special policies. We learned valuable lessons operating this system since around 2021.

CentOS 7 End of Life

One of the driving forces was the end of support for CentOS 7. To condense a really long discussion, we looked at the drama around the RHEL rebuilders and opted to install Debian as the base OS for most systems unless it was directly used as a platform by users (or it was otherwise unfeasible). For these we would use Alma Linux 9.

We really did not want to carry Torque into the new era on Alma Linux 9 so there was pressure to implement HTCondor as soon as possible. It turned out that we managed to do it only for Stoomboot; the Grid is still a work in progress.

Well, master, we're in a fix and no mistake

There was mounting pressure to get everything done in time: the upgrades for all our CentOS 7 systems; the installation of HTCondor on Debian and the move of Stoomboot.

We managed in the end but it was not a pleasant experience.

The Plan

On the considerations that went into the transition.

Goals

We set out to make the transition as smooth as possible for our users. While some users were already familiar with HTCondor on other systems (e.g. CERN) we wanted to give our Torque addicts at least some sense of familiarity.

Towards danger; but not too rashly, nor too straight

We started setting up a brand new cluster with only a few worker nodes. We invited some early adopters to be our guinea pigs and validate the basic setup. Then we sneakily started to move more and more worker nodes over to the new system and suggested to our users to join the nice newer (and larger) cluster. A freshly added set of worker nodes with 128 core AMD Bergamo CPUs was a great lure.

Communication and Support

During the transition phase we spend more time on user support via various channels. Luckily most of the users found the transition quite easy and were reasonably happy with the new system.

Turning off the Old System

At some point we had to announce that the old system was going to be turned off. We allowed the last stragglers to finish their work so they did not have to make all kind of late changes in their workflows.

Contact with HTCondor Central

We received great support from the HTCondor team in Wisconsin. A timely visit there, on the heels of an in-person LIGO meeting, was especially useful to hammer out the policies and settings.

A Far Green Country under a Swift Sunrise

When we started the process of implementing a local facility under HTCondor control, the mindset was very much on the classic way of thinking about batch systems. The consideration was that operators and users were already familiar with a certain way of working and the transition would go much more smoothly if we could shape the new system to look like the old.

This meant that we had to introduce a concept that is not native to HTCondor: a limited wall time. This can be done through a job transformation.

Users set the `MaxWallTime` (directly, or via a job category) and we add a `SYSTEM_PERIODIC_HOLD` condition if this gets exceeded.

Re-imagining the queues

We therefore introduced the concept of job categories (a bad name indeed!) and used a transformation to enforce the choice of a category and implement placing the job on hold once it overran the specified duration.

```
JOB_TRANSFORM_NAMES = $(JOB_TRANSFORM_NAMES) JobCategoryNotDefined JobCategorySetDefaults \
    SetMaxWallTime JobCategoryChecks
SUBMIT_REQUIREMENT_NAMES = $(SUBMIT_REQUIREMENT_NAMES) JobCategoryNotDefinedCheck JobCategoryInvalidCheck \
    JobCategoryMaxWallTimeCheck
SCHEDD_CLASSAD_USER_MAP_NAMES = $(SCHEDD_CLASSAD_USER_MAP_NAMES) JobCategoryDefaultWallTime \
    JobCategoryMaxWallTime
SYSTEM_PERIODIC_HOLD_NAMES = $(SYSTEM_PERIODIC_HOLD_NAMES) JobCategoryWallTime
SYSTEM_PERIODIC_REMOVE_NAMES = $(SYSTEM_PERIODIC_REMOVE_NAMES) JobMaxHoldTime

JOB_TRANSFORM_JobCategoryNotDefined @=end
    NAME JobCategoryNotDefined
    REQUIREMENTS (JobUniverse =?= 5 && isUndefined(JobCategory))
    SET JobCategoryNotDefined TRUE
@end
SUBMIT_REQUIREMENT_JobCategoryNotDefinedCheck = isUndefined(JobCategoryNotDefined)
SUBMIT_REQUIREMENT_JobCategoryNotDefinedCheck_REASON = "+JobCategory must be defined, see https://__knowledge_base_url__"


# Default Walltime (in seconds) per job category
CLASSAD_USER_MAPDATA_JobCategoryDefaultWallTime  @=end
* express     600
* short    4*3600
* medium  24*3600
* long    96*3600
@end

# Maximum Walltime (in seconds) per job category
CLASSAD_USER_MAPDATA_JobCategoryMaxWallTime  @=end
* express     600
* short    4*3600
* medium  24*3600
* long    96*3600
@end
```

```
JOB_TRANSFORM_JobCategorySetDefaults @=end
    NAME JobCategorySetDefaults
    REQUIREMENTS (JobUniverse =?= 5 && !isUndefined(JobCategory))
    EVALSET JobCategoryDefaultWallTime eval(userMap("JobCategoryDefaultWallTime", toLower(JobCategory)))
    EVALSET JobCategoryMaxWallTime eval(userMap("JobCategoryMaxWallTime", toLower(JobCategory)))
    EVALSET JobCategoryOK (isError(JobCategoryDefaultWallTime) || isError(JobCategoryMaxWallTime)) ? FALSE : TRUE
@end
SUBMIT_REQUIREMENT_JobCategoryInvalidCheck = JobUniverse =?= 5 ? JobCategoryOK : TRUE
SUBMIT_REQUIREMENT_JobCategoryInvalidCheck_REASON = "Invalid JobCategory, see https://__knowledge_base_url__"

JOB_TRANSFORM_SetMaxWallTime @=end
    NAME SetMaxWallTime
    REQUIREMENTS (JobUniverse =?= 5 && isUndefined(MaxWallTime))
    EVALSET MaxWallTime JobCategoryDefaultWallTime
@end

JOB_TRANSFORM_JobCategoryChecks @=end
    NAME SetMaxWallTime
    REQUIREMENTS JobUniverse =?= 5
    EVALSET JobCategoryMaxWallTimeOK MaxWallTime > JobCategoryMaxWallTime ? FALSE : TRUE
@end
SUBMIT_REQUIREMENT_JobCategoryMaxWallTimeCheck = JobUniverse =?= 5 ? JobCategoryMaxWallTimeOK : TRUE
SUBMIT_REQUIREMENT_JobCategoryMaxWallTimeCheck_REASON = strcat("MaxWallTime must be less than or equal to \
    the job category (", JobCategory, ") maximum walltime of ", JobCategoryMaxWallTime, " seconds")

# hold running jobs that exceed their max walltime
SYSTEM_PERIODIC_HOLD_JobCategoryWallTime = JobStatus == 2 && time() - EnteredCurrentStatus > MaxWallTime
SYSTEM_PERIODIC_HOLD_JobCategoryWallTime_REASON = strcat("job exceed MaxWallTime of ", MaxWallTime)

# delete held jobs after one week
SYSTEM_PERIODIC_REMOVE_JobMaxHoldTime = JobStatus == 5 && time() - EnteredCurrentStatus > 7 * 24 * 3600
SYSTEM_PERIODIC_REMOVE_JobMaxHoldTime_REASON = "job removed after being held for 1 week"
```

Everything in a Container

The time table for the move to HTCondor coincided with the end of life for CentOS 7. We were worried that some users would not be prepared to port their work to a newer operating system and therefore we opted to run everything in containers from the get go.

Because we run everything in a container, the user is not going to notice that they are on a Debian system unless they have a close look at the kernel.

HTCondor has support for singularity (or Apptainer) and we went with three flavours of Enterprise Linux derivatives: el7, el8, and el9. We also allow users to bring their own container image.

The following job transformation takes care of setting the SingularityImage accordingly.

Absurdly simple, like most riddles when you see the answer

```
# Submit Requirement: ValidSingularityJob

SUBMIT_REQUIREMENT_NAMES = $(SUBMIT_REQUIREMENT_NAMES) NotDefinedOSorImageCheck DefinedOSandImageCheck \
    SetSingularityImageFromOSCheck ValidSingularityImage

SUBMIT_REQUIREMENT_ValidSingularityImage = JobUniverse =?= 5 ? (!isUndefined(SingularityImage) && \
    !isError(SingularityImage)) : TRUE

SCHEDD_CLASSAD_USER_MAP_NAMES = $(SCHEDD_CLASSAD_USER_MAP_NAMES) Trust JobImages JobOS

JOB_TRANSFORM_NAMES = $(JOB_TRANSFORM_NAMES) NotDefinedOSorImage DefinedOSandImage SetSingularityImageFromOS

# Handle container image
CLASSAD_USER_MAPDATA_JobImages @=end
* el7 /cvmfs/unpacked.cern.ch/gitlab-registry.cern.ch/sft/docker/centos7-core:latest
* el8 /cvmfs/unpacked.cern.ch/gitlab-registry.cern.ch/sft/docker/alma8-core:latest
* el9 /cvmfs/unpacked.cern.ch/gitlab-registry.cern.ch/sft/docker/alma9-core:latest
@end

CLASSAD_USER_MAPDATA_JobOS @=end
* el7 CentOS7
* el8 CentOS8
* el9 AlmaLinux9
@end
```

```
JOB_TRANSFORM_NotDefinedOSorImage @=end
    NAME NotDefinedOSorImage
    REQUIREMENTS (JobUniverse =?= 5 && isUndefined(SingularityImage) && isUndefined(UseOS))
    SET NotDefinedOSorImage TRUE
@end
SUBMIT_REQUIREMENT_NotDefinedOSorImageCheck = isUndefined(NotDefinedOSorImage)
SUBMIT_REQUIREMENT_NotDefinedOSorImageCheck_REASON = "Either set +UseOS = \"el7\" or \"el8\" or \"el9\"; or \
    set +SingularityImage = (path to container image file or unpackaged container dir)"

JOB_TRANSFORM_DefinedOSandImage @=end
    NAME DefinedOSandImage
    REQUIREMENTS (JobUniverse =?= 5 && !isUndefined(SingularityImage) && !isUndefined(UseOS))
    SET DefinedOSandImage TRUE
@end
SUBMIT_REQUIREMENT_DefinedOSandImageCheck = isUndefined(DefinedOSandImage)
SUBMIT_REQUIREMENT_DefinedOSandImageCheck_REASON = "Specify either +UseOS or +SingularityImage not both"

JOB_TRANSFORM_SetSingularityImageFromOS @=end
    NAME SetSingularityImageFromOS
    REQUIREMENTS (JobUniverse =?= 5 && isUndefined(SingularityImage) && !isUndefined(UseOS))
    EVALSET MappedImage userMap("JobImages", toLower(UseOS))
    EVALSET SingularityImage (MappedImage =?= Undefined) ? Error : MappedImage
    SET SetSingularityImageFromOS TRUE
@end
SUBMIT_REQUIREMENT_SetSingularityImageFromOSCheck = isUndefined(SetSingularityImageFromOS) ? \
    TRUE : !isError(SingularityImage)
SUBMIT_REQUIREMENT_SetSingularityImageFromOSCheck_REASON = "The requested OS is not valid. Set \
    +UseOS = \"el7\" or \"el8\" or \"el9\""
```

Accounting Groups

One of the more complicated tweaks of the original system was that we accounted users by their groups.

For instance the alice group is treated as a single entity for the sake of the use of their share, so one particularly productive user could take a considerable chunk out of that.

Special Treatment

To complicate matters further, we have some groups that have contributed entire subclusters through their funding. This entitles them to have priority use on the hardware they paid for.

In the transition to HTCondor we loosened the ties to the specific hardware somewhat, because there is nothing special about one worker node or another; a cycle is a cycle. We acknowledge their entitlement to a certain amount of benchmark units that is equivalent to what the hardware provides.

We still struggle to define a meaningful configuration that would reflect the respective priorities for these groups.

```
SUBMIT_REQUIREMENT_NAMES = $(SUBMIT_REQUIREMENT_NAMES) ValidAcctGroup
SCHEDD_CLASSAD_USER_MAP_NAMES = $(SCHEDD_CLASSAD_USER_MAP_NAMES) PosixGroups QuotaGroups
JOB_TRANSFORM_NAMES = $(JOB_TRANSFORM_NAMES) ValidAcctGroup SetAccountingGroupDefault SetAccountingGroupRequested

CLASSAD_USER_MAPFILE_PosixGroups = /etc/condor/user-group.map

# Groups with quotas, plus datagrid for testing
CLASSAD_USER_MAPDATA_QuotaGroups @=end
* smefit    smefit
* gravwav   gravwav
* datagrid  datagrid
@end
```

```
# Check if the user is a member of the posixgroup that they requested
# via accounting_group (which is the AcctGroup Classad)
JOB_TRANSFORM_ValidAcctGroup @=end
    NAME ValidAcctGroup
    REQUIREMENTS (JobUniverse =?= 5 || JobUniverse =?= 7) && !isUndefined(AcctGroup)
    EVALSET PosixGroup = userMap("PosixGroups", Owner, AcctGroup)
    EVALSET isMemberofPosixGroup = PosixGroup == AcctGroup
@end
SUBMIT_REQUIREMENT_ValidAcctGroup = isUndefined(AcctGroup) || isMemberofPosixGroup
SUBMIT_REQUIREMENT_ValidAcctGroup_REASON = "Invalid accounting_group set. accounting_group must \
   either be undefined, or set to a posix group that you are a member of."

JOB_TRANSFORM_SetAccountingGroupDefault @=end
   NAME SetAccountingGroupDefault
   REQUIREMENTS (JobUniverse =?= 5 || JobUniverse =?= 7) && isUndefined(AcctGroup)
   EVALSET AcctGroupUser = Owner
   EVALSET AccountingGroup = Owner
@end

JOB_TRANSFORM_SetAccountingGroupRequested @=end
   NAME SetAccountingGroupRequested
   REQUIREMENTS (JobUniverse =?= 5 || JobUniverse =?= 7) && !isUndefined(AcctGroup)
   EVALSET isQuotaGroup = userMap("QuotaGroups", AcctGroup)
   EVALSET AcctGroupUser = Owner
   EVALSET AccountingGroup = isUndefined(isQuotaGroup) ? Owner : join(".", AcctGroup, AcctGroupUser)
@end

IMMUTABLE_JOB_ATTRS = $(IMMUTABLE_JOB_ATTRS) AccountingGroup
```

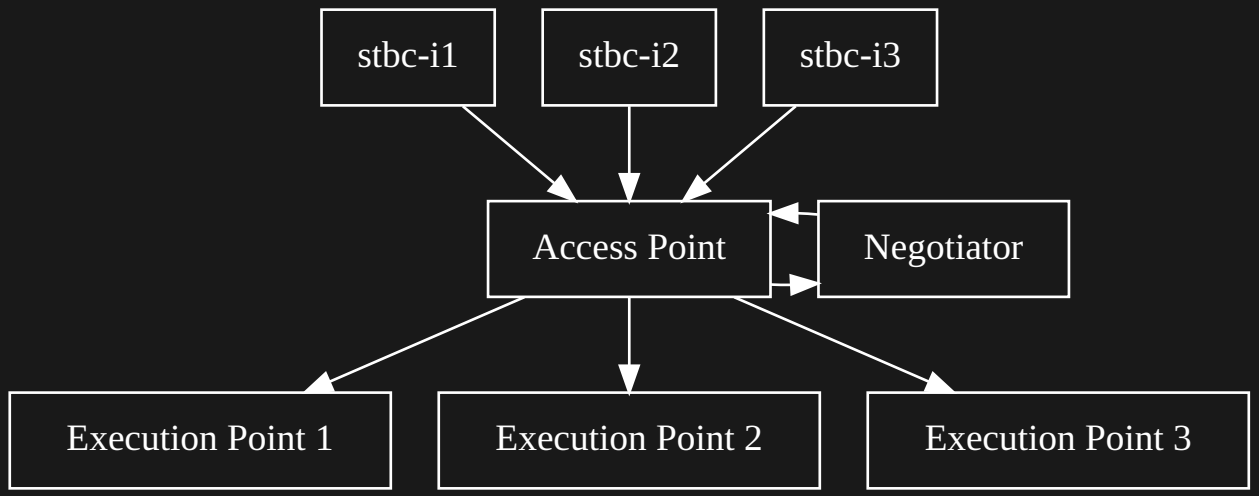
The Express Negotiator

We've set aside a machine whose slots will only go to jobs with a very short deadline. Basically these are only going to be test jobs. By combining two negotiators, one for express jobs and one for everything else, we make sure these express jobs make it to that one node.

```
# regular negotiator
NEGOTIATOR_SLOT_CONSTRAINT = !regexp("wn-sate-079", Machine)

# express negotiator
NEGOTIATOR_SLOT_CONSTRAINT = regexp("wn-sate-079", Machine)
# Match only specific jobs
NEGOTIATOR_JOB_CONSTRAINT = MaxWallTime < 600
```

Layout of nodes



Access Point

We have a single access point on an old worker node. The choice for a bit of beefy hardware was due to the expected load on the machine. We do not allow users to log on to the AP at all, but they can reach it from any of the interactive nodes.

The Road Goes Ever On

Now that our local cluster has successfully made the transition to HTCondor, it is time to look at the Grid cluster. Although the parameters are different, we've built up enough confidence with HTCondor that we should be able to set up a cluster to suit our needs. But work on this has only just begun.

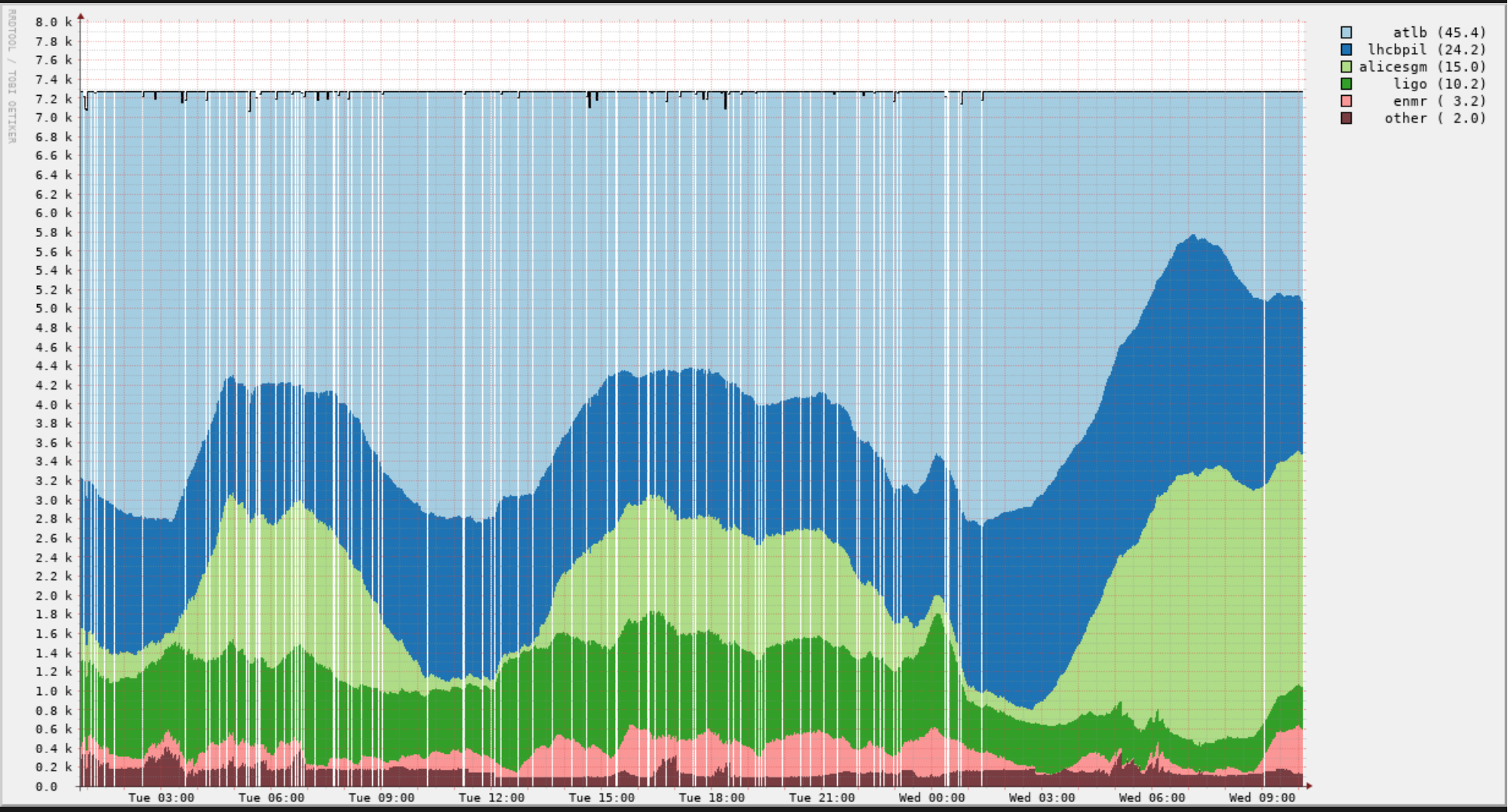
The advantage is that users won't notice anything at all, as the front end system (ARC-CE) stays the same.

The Cracks of Doom

We did not want to carry our ageing Torque system to a newer platform, but we ended up doing this for the Grid anyway because there was no time to implement HTCondor for grid before CentOS7 EOL.

It turns out that Torque would run on Alma Linux 9, but it is beginning to show some cracks.

We will continue working on making this transition.


RRDTOOL / TOBI OETIKER
8.0 k
7.8 k
7.6 k
7.4 k
7.2 k
7.0 k
6.8 k
6.6 k
6.4 k
6.2 k
6.0 k
5.8 k
5.6 k
5.4 k
5.2 k
5.0 k
4.8 k
4.6 k
4.4 k
4.2 k
4.0 k
3.8 k
3.6 k
3.4 k
3.2 k
3.0 k
2.8 k
2.6 k
2.4 k
2.2 k
2.0 k
1.8 k
1.6 k
1.4 k
1.2 k
1.0 k
0.8 k
0.6 k
0.4 k
0.2 k
0.0
Tue 03:00
Tue 06:00
Tue 09:00
Tue 12:00
Tue 15:00
Tue 18:00
Tue 21:00
Wed 00:00
Wed 03:00
Wed 06:00
Wed 09:00
atlb (45.4)
lhcbpil (24.2)
alicesgm (15.0)
ligo (10.2)
enmr (3.2)
other (2.0)

Better Implementation of Priorities

We are still tweaking the Stoomboot system. The system that schedules and prioritises jobs from different users and groups needs some work.

He that breaks a thing to find out what it is has left the path of wisdom.

— Gandalf the Grey

Go not to the HTCondor developers for counsel, for they will say both no and yes.

— Frodo